Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. In a system having multiple processors wherein each processor enters a debug mode of operation as a result of incurring a debug event, an improvement comprising:

a debug interface associated with each one processor which asserts a debug event signal when the associated one processor enters the debug mode as a result of incurring an internal debug event; and

a logic circuit associated with each processor and receptive of the debug event signal supplied by each other processor of the system, each logic circuit responding to the debug event signal supplied by each other processor to assert an external debug break signal to the processor associated with the logic circuit; and

the debug interface further responds to the external debug break signal supplied by the associated logic circuit to place the processor into the debug mode.

2. In a system as defined in claim 1 wherein:

each logic circuit asserts the external debug break signal to its associated processor approximately simultaneously with the assertion of the debug event signal.

3. In a system as defined in claim 1 wherein:

the logic circuit associated with one processor further responds to the assertion of the debug event signal by the one processor to prevent the assertion of the external debug break signal to the one processor.

4. In a system as defined in claim 1 wherein:

each logic circuit is further responsive to a debug reset signal to de-assert the external debug break signal upon the assertion of the debug reset signal.

5. In a system as defined in claim 1 wherein each logic circuit further includes:

an OR gate connected to each other processor to receive the debug event signal asserted by any one processor;

an AND gate connected to the OR gate, the AND gate receiving a signal indicative of the assertion of the debug event signal by the associated processor; and wherein:

the OR gate supplies a signal to the AND gate indicative of the assertion of a debug event signal from any other processor; and

the AND gate supplies the external debug break signal to the associated processor in response to the assertion of the debug event signal from any other processor; and

the AND gate de-asserts the external debug break signal to the associated processor in response to the negation of the debug event signal from any other processor.

6. In a system as defined in claim 5, wherein:

a debug reset signal is asserted to remove all other processors from the debug mode after the one processor ceases operation in the debug mode; and

the AND gate receives the debug reset signal as an additional input signal to de-assert the external debug break signal to the associated processor upon the assertion of the debug reset signal.

7. In a system as defined in claim 5, wherein each logic circuit further includes:

an inverter connected to receive the debug event signal from the associated processor and operative to invert the debug event signal and supply an inverted debug event signal as one input signal to the AND gate.

8. In a system as defined in claim 7, wherein:

the inverted debug event signal causes the AND gate to supply the external debug break signal only when the debug event signal is not asserted by the one processor.

9. In a system as defined in claim 1, wherein the debug interface comprises:

debug prioritization logic receptive of signals indicative of external debug break signals and indicative internal debug events incurred by the associated processor, the debug prioritization logic placing the associated processor into the debug mode of operation in response to signals indicative of either the external debug break signals or the internal debug events.

10. In a system as defined in claim 9, wherein the debug interface further comprises:

a register connected to receive the signal from the debug prioritization logic indicative of whether the processor was placed into the debug mode of operation in response to either the external debug break signal or the internal debug events.

11. In a system as defined in claim 9, wherein the debug interface further comprises:

a flip-flop receptive of the external debug break signal and connected to the debug prioritization logic, the flip-flop supplying a trigger debug break signal to the debug prioritization logic in response to the external debug break signal.

12. In a system as defined in claim 9, wherein each processor executes instructions in a pipeline, and the debug interface further comprises:

a pipeline flush mechanism connected to the debug prioritization logic and responsive to flush instructions from the pipeline prior to assertion of the debug event signal.

13. A method of approximately simultaneously halting execution of instructions by each processor of a system having multiple processors upon one of

the processors of the system entering a debug mode of operation as a result of incurring a debug event, comprising:

halting execution of instructions by the one processor to place that one processor into a debug mode of operation upon that one processor incurring a debug event; and

after halting execution of instructions by the one processor, placing each other processor of the system into a debug mode of operation approximately simultaneously with the one processor entering the debug mode of operation by sending an external debug break signal from the one processor to each other processor of the system to cause each other processor to enter the debug mode of operation.

14. A method as defined in claim 13, further comprising:

asserting a debug event signal from the one processor upon entering the debug mode of operation; and

placing each other processor into the debug mode in response to the assertion of the debug event signal by the one processor by generating the external debug break signals for the other processors of the system in response to the assertion of the debug event signal by the one processor.

15. A method as defined in claim 13, further comprising:

placing each other processor into the debug mode by the assertion of the external debug break signal to each other processor;

asserting the external debug break signal to each other processor in response to the one processor asserting ~~the~~ a debug event signal.

16. A method as defined in claim 15, further comprising:

inhibiting the assertion of the external debug break signal to a processor which is asserting the debug event signal.

17. A method as defined in claim 13, further comprising:

utilizing a logic circuit associated with each processor to generate an external ~~debut~~ debug break signal for the associated processor; and

applying ~~the~~ a debug event signal from the one processor to the logic circuit associated with each other processor.

18. A method as defined in claim 13, further comprising:

connecting an OR gate to each other processor to receive a signal indicative of the assertion of the debug event signal by the one processor;

connecting an AND gate to the OR gate;

supplying a signal from the OR gate to the AND gate indicative of the occurrence of a debug event signal from the one processor;

supplying the external debug break signal from the AND gate to the associated processor in response to the occurrence of a debug event signal from the one processor; and

de-asserting the external debug break signal to the associated processor in response to the de-assertion of the debug event signal from the one processor.

19. A method as defined in claim 18, further comprising:

asserting a debug reset signal to remove all other processors from the debug mode after the one processor ceases operation in the debug mode.

20. A method as defined in claim 19, further comprising:

supplying the debug reset signal as an additional input signal to the AND gate; and

de-asserting the external debug break signal upon the assertion of the debug reset signal.

21. A method as defined in claim 20, further comprising:

connecting an inverter to receive and invert the debug event signal from the associated processor; and

supplying the inverted debug event signal as an input signal to the AND gate.

22. A method as defined in claim 21, further comprising:

inverting the debug event signal to a logical level which causes the AND gate to supply the external debug break signal to the associated processor only when the debug event signal is not asserted by the associated processor.

23. A method as defined in claim 13, further comprising:

applying a signal indicative of the external debug break signal and a signal indicative of an internal debug event incurred by the associated processor to debug prioritization logic of the associated processor; and

placing the associated processor into the debug mode of operation in response to signals indicative of either the external debug break signal or an internal debug event.

24. A method as defined in claim 23, further comprising:

indicating in a register connected to the debug prioritization logic whether the processor was placed into the debug mode of operation in response to either the external debug break signal or the internal debug event.

25. A method as defined in claim 23, further comprising:

flushing instructions from a pipeline of the associated processor in response to the associated processor incurring an internal debug event and before the associated processor asserts the debug event signal.